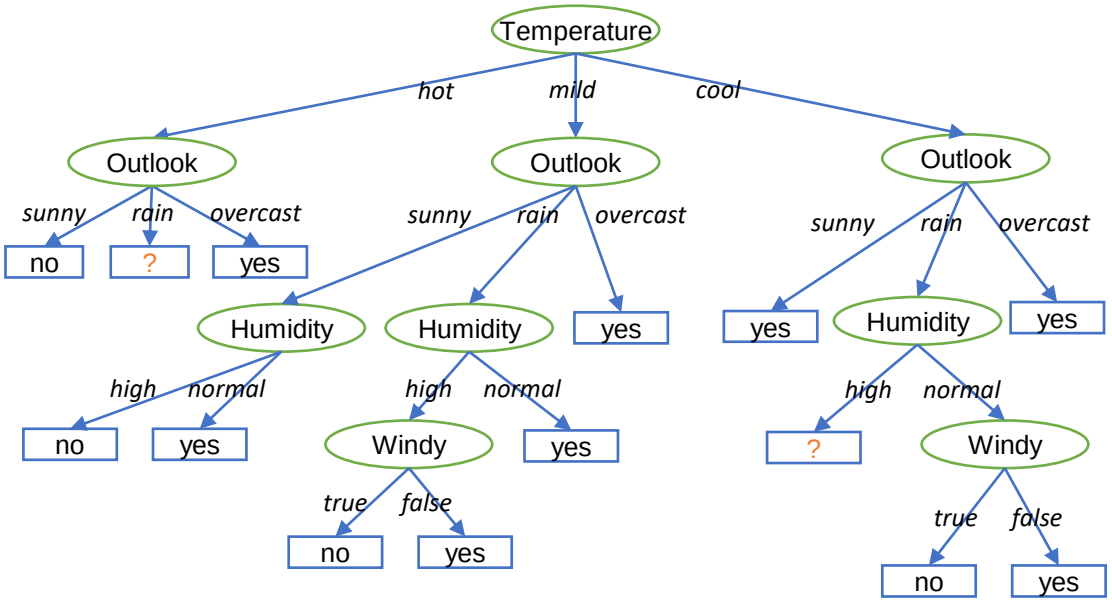
Decision Tree

A Decision Tree is a tree with nodes representing deterministic decisions based on variables and edges representing path to next node or a leaf node based on the decision

Temperature	Outlook	Humidity	Windy	Play?
hot	sunny	high	false	no
hot	sunny	high	true	no
hot	overcast	high	false	yes
cold	rain	normal	false	yes
cold	overcast	normal	true	yes
mild	sunny	high	false	no
cold	sunny	normal	false	yes
mild	rain	normal	false	yes
mild	sunny	normal	true	yes
mild	overcast	high	true	yes
hot	overcast	normal	false	yes
mild	rain	high	true	no



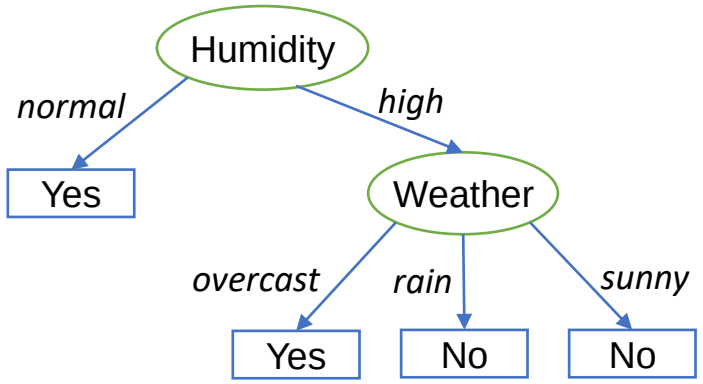
Decision Tree

Straightforward construction of *decision tree* leads that tree size **exponentially** depends on input data.

At the worst-case scenario n categorical features with **m** possible (*without taking in to account continuous variables!*) values each will have $O(m^n)$, for binary features, when $m = 2$ it is $\boldsymbol{O(2^n)}$.

A Decision Tree is modelled on a **simple series of questions** that lead serially to an answer that best fits the data used in training.

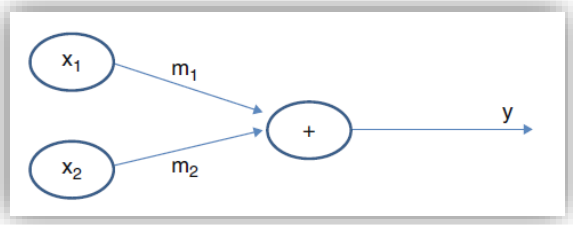
Temperature	Outlook	Humidity	Windy	Play?
hot	sunny	high	false	no
hot	sunny	high	true	no
hot	overcast	high	false	yes
cold	rain	normal	false	yes
cold	overcast	normal	true	yes
mild	sunny	high	false	no
cold	sunny	normal	false	yes
mild	rain	normal	false	yes
mild	sunny	normal	true	yes
mild	overcast	high	true	yes
hot	overcast	normal	false	yes
mild	rain	high	true	no



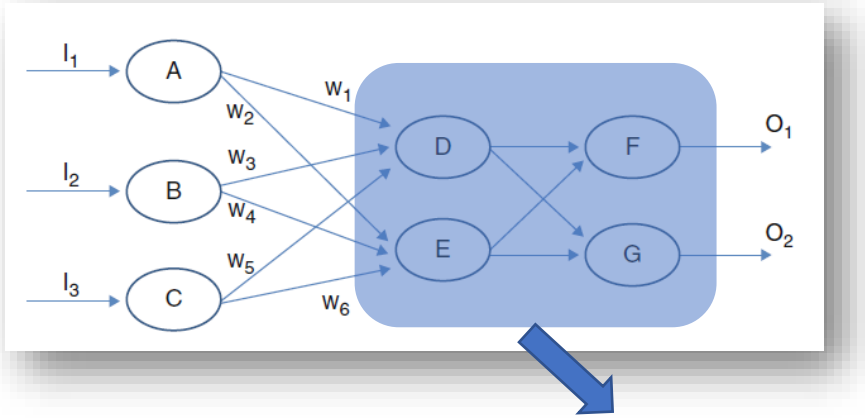
Visualizing ANN Equations

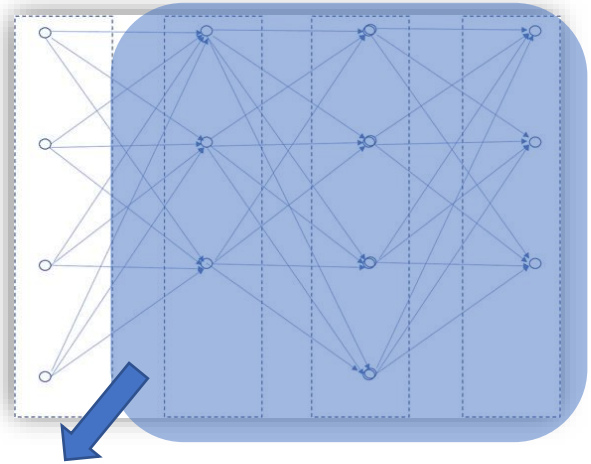
Graph, representing:

$$y = m_1 * x_1 + m_2 * x_2$$



Graphs representation of a sample neural networks





nodes contains activation functions as in logistic regression

Designing and training ANN

1. Numbers of layers
2. Number of nodes in each Layer
3. Level of nodes connectivity
4. Activation functions used in each node
5. Does the weights are shared between multiple connections (CNN)
6. Feedbacks (RNN)
7. Loss (cost) function

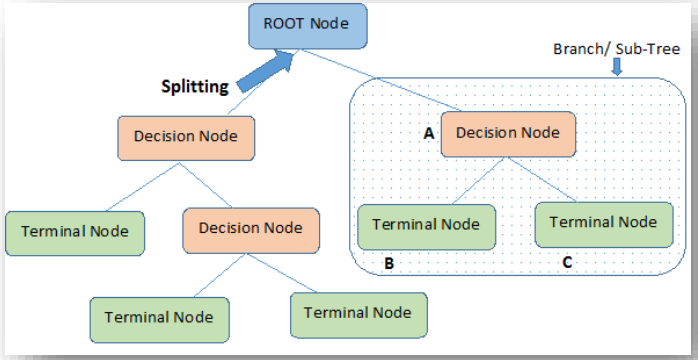
Characteristics defined of ANN size, complexity, architecture (***hyperparameters***)

Choose a set of hyperparameters (such as number of layers, the activation functions etc)
begin loop // Iterate 10s to 100s of times depending on problem and time constraints
Train the neural network model for correct weight (details later)
If the loss function is low – you have a trained network and exit the loop
If the loss function is high – modify your set of hyperparameters
end loop

Important Terminology related to Decision Trees

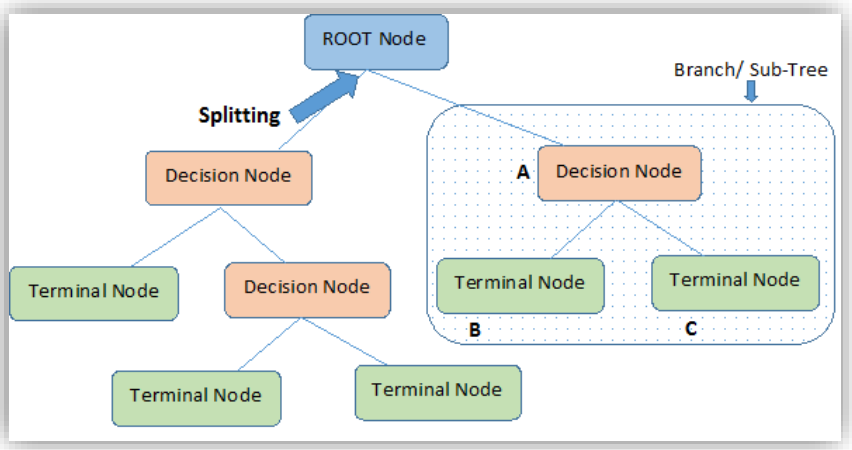
- **Root Node:** It represents entire population or sample and this further gets divided into two or more sets.

- **Splitting:** It is a process of dividing a node into two or more sub-nodes.
- **Decision Node:** When a sub-node splits into further sub-nodes, then it is called decision node.
- **Leaf / Terminal Node:** Nodes do not split is called Leaf or Terminal node.



Important Terminology related to Decision Trees

- **Pruning:** When we remove sub-nodes of a decision node, this process is called pruning.
- **Branch / Sub-Tree:** A sub section of entire tree is called branch or sub-tree.



Decision tree

Parameters: training data, decision metric

Step 1: Build the root with variables of most importance

Step2: Build a decision of **highest information split**

Step3: Recursively construct the nodes and decision using 1 and 2 step until no information can be split on the edge node

Decision tree

Decision metrics for classification:

- Misclassification error
- Gini Index
- Cross-entropy or deviance

Decision metrics for regression:

- Squared residuals minimization algorithm which implies that expected sum variances for two resulting nodes should be minimized.

$$\underset{x_j \le x_j^R, j=1,\ldots,M}{\operatorname{argmin}} \; [P_l Var(Y_l) + P_r Var(Y_r)]$$

Y_l, Y_r - response vectors for corresponding left and right child nodes

P_l, P_r - respective frequencies

Decision tree

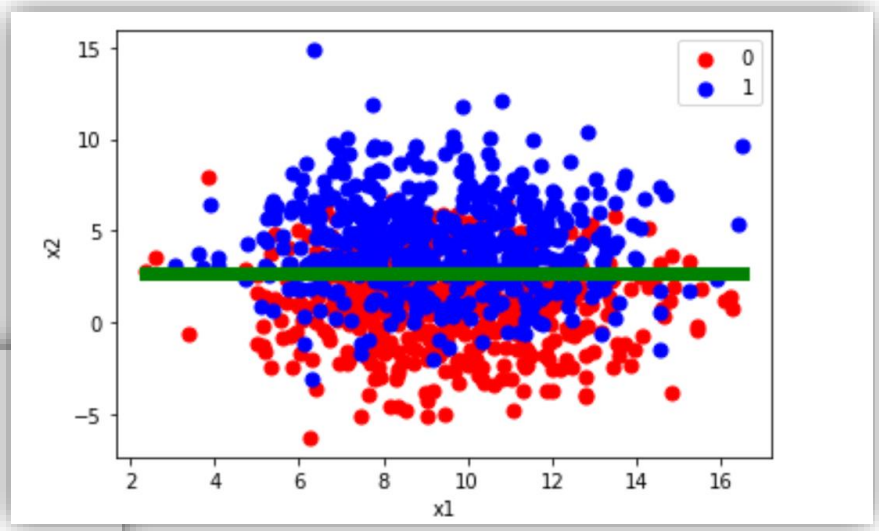
Gini impurity is a measure of how often a randomly chosen element from the set is incorrectly labelled if it were labelled according to the distribution of labels in the subset.

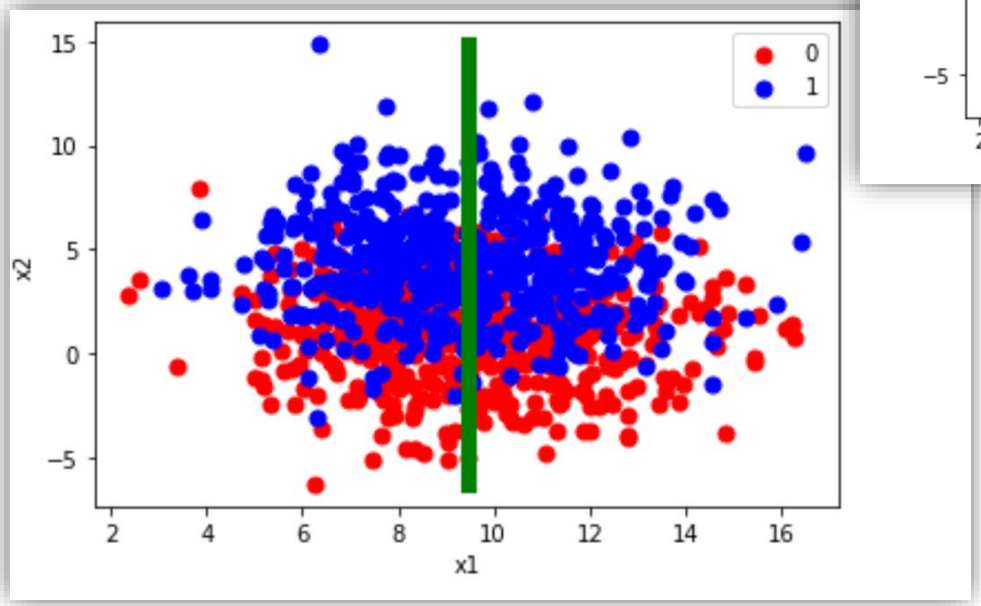
$$Ic(p) = \sum_{i=0}^{J} \left(p_i * \sum_{k \neq i}^{J} p_k \right) = \sum_{i=0}^{J} p_i(1 - p_i) = 1 - \sum_{i=0}^{J} {p_i}^2$$

Higher Gini impurity refers to higher chance of misclassification conversely, and lower impurity refers to lower chance of misclassification

In constructing a Decision Tree split, the goal is to split with the lowest weighted Gini impurity value for the child trees.

SplittingCriteria.ipynb





Entropy

Less impure node requires less information to describe it. And, more impure node requires more information.

Using information theory, we estimate the amount of information contained within each variable. A key measure in information theory is **entropy**.

$$H = -\sum_{i=1}^{n} (p_i \log_2 p_i) \Longrightarrow$$

$\boldsymbol{p_i}$ is probability of occurrence of $\boldsymbol{i}$ -th possible value
$\boldsymbol{n}$ is the number of values
$\boldsymbol{H}$ is measure of entropy

Dice with possible values of 1–6

$$H = -\sum_{i=1}^{6} \left(\frac{1}{6} \log \frac{1}{6}\right) \approx 2.58$$

Flip coin

$$H = -\sum_{i=1}^{2} \left(\frac{1}{2} \log \frac{1}{2}\right) \approx 1$$

Entropy

Variable 1	Variable 2	Outcome
3	5	Stop
7	6	Continue
3	3	Stop
4	8	Continue
3	9	Continue
6	5	Stop
5	8	Continue
6	4	Continue

$$H(outcome) = -\left(\frac{3}{8}\log\frac{3}{8}\right) - \left(\frac{5}{8}\log\frac{5}{8}\right) \approx 0.954$$

Decision on Variable 1: >**4 (greater than 4)**

Decision on Variable 2: >**6 (greater than 6)**

*based on variable average **

Entropy

$$H(outcome) = -\left(\frac{3}{8}\log\frac{3}{8}\right) - \left(\frac{5}{8}\log\frac{5}{8}\right) \approx 0.954$$

Entropy before taking decision

$$H(> 4, variable1) = -\left(\frac{3}{4}\log\frac{3}{4}\right) - \left(\frac{1}{4}\log\frac{1}{4}\right) \approx 0.81$$

$$H(\leq 4, variable1) = -\left(\frac{2}{4}\log\frac{2}{4}\right) - \left(\frac{2}{4}\log\frac{2}{4}\right) \approx 1$$

$$H(> 6, variable2) = -\left(\frac{3}{3}\log\frac{3}{3}\right) - \left(\frac{0}{3}\log\frac{0}{3}\right) = 0$$

$$H(\leq 6, variable2) = -\left(\frac{2}{5}\log\frac{2}{5}\right) - \left(\frac{3}{5}\log\frac{3}{5}\right) = 0.97$$

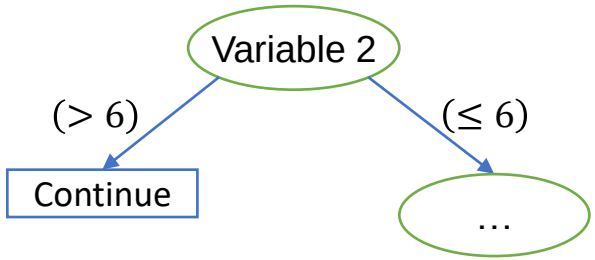
$$H(outcome, variable1) = -p_{>4}H(> 4) - p_{\leq 4}H(\leq 4) = \left(\frac{4}{8}\right) * 0.81 + \left(\frac{4}{8}\right) * 1 = 0.9$$

$$H(outcome, variable2) = -p_{>6}H(> 6) - p_{\leq 6}H(\leq 6) = \left(\frac{3}{8}\right) * 0 + \left(\frac{5}{8}\right) * 0.97 = 0.61$$

Entropy after decision

$$IG = H(outcome) - H(outcome, variable1) = 0.954 - 0.9 = 0.054$$

$$IG = H(outcome) - H(outcome, variable2) = 0.954 - 0.61 = 0.344$$



Exercise 2

Temperature	Outlook	Humidity	Windy	Play?
hot	sunny	high	false	no
hot	sunny	high	true	no
hot	overcast	high	false	yes
cold	rain	normal	false	yes
cold	overcast	normal	true	yes
mild	sunny	high	false	no
cold	sunny	normal	false	yes
mild	rain	normal	false	yes
mild	sunny	normal	true	yes
mild	overcast	high	true	yes
hot	overcast	normal	false	yes
mild	rain	high	true	no

$$IG = H(play) - H(play, temperature) = \_\_\_$$

$$IG = H(play) - H(play, outlook) = \_\_\_$$

$$IG = H(play) - H(play, windy) = \_\_\_$$

8 minutes

Termination conditions. Pre-pruning

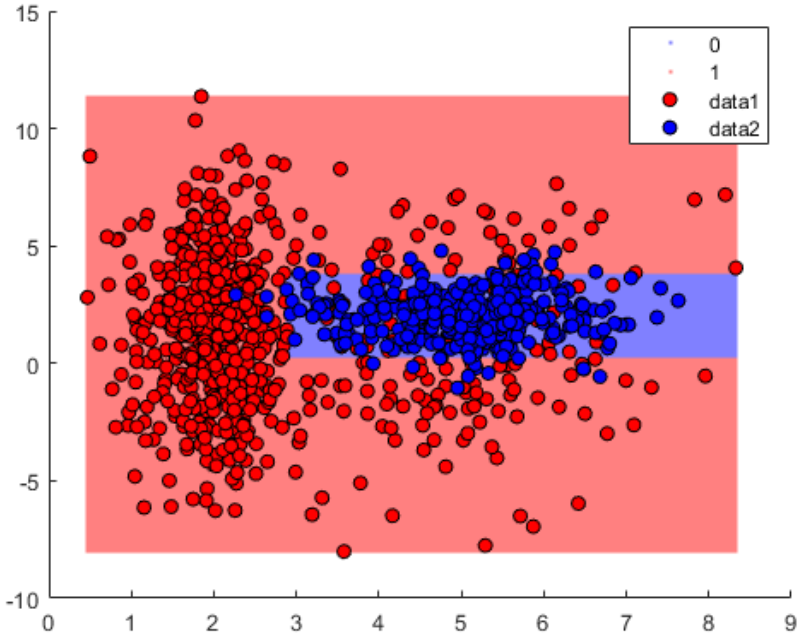
- Maximum number of instances in node
- Maximum number of splits
- Maximum number of leaves
- Maximum depth of a tree
- …

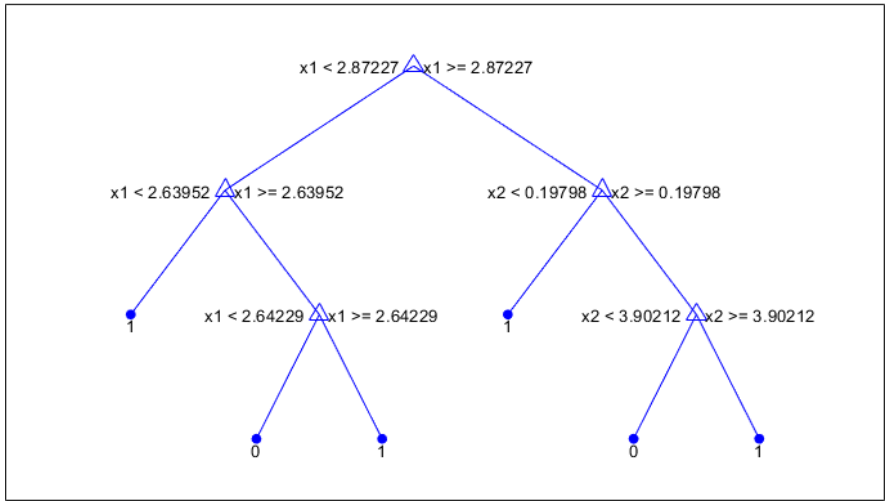
Post-pruning

Used after the maximum decision tree is constructed.

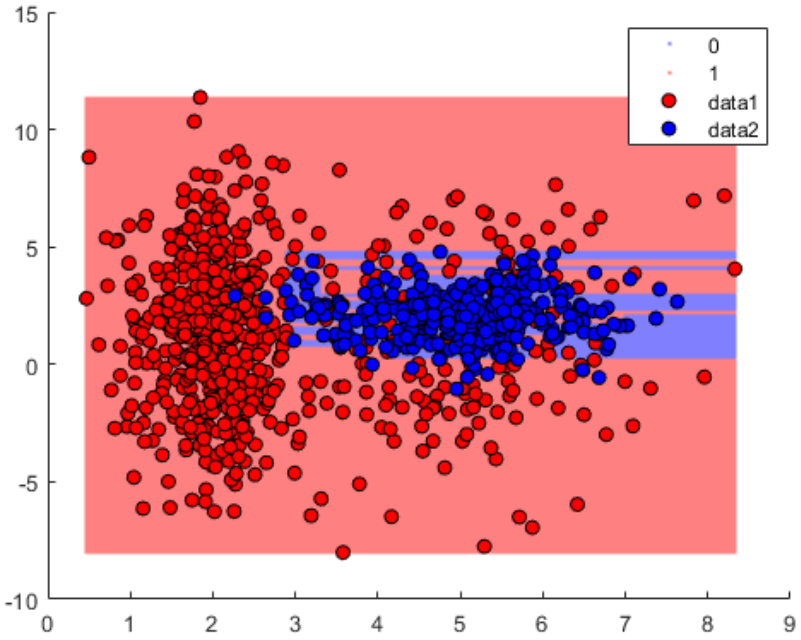
- Misclassification rate
- Minimum-error pruning
- Pessimistic error pruning

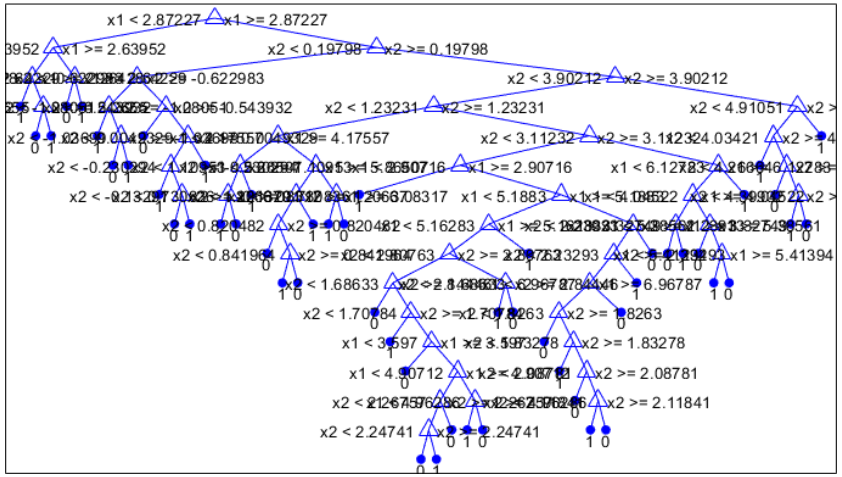
Decision tree



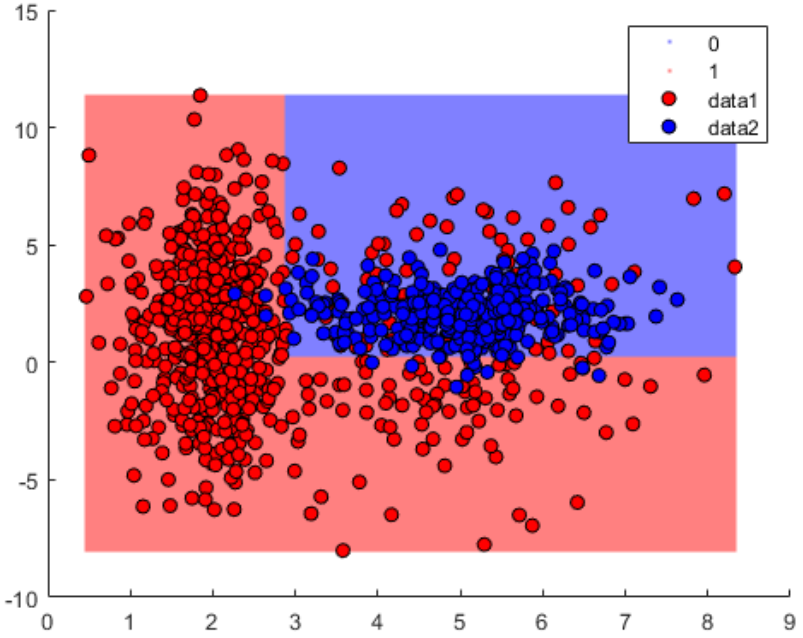


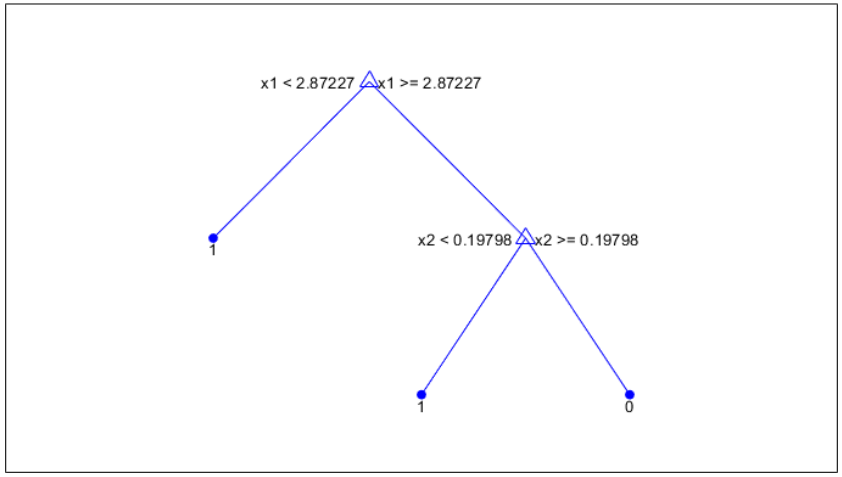
Decision tree





Decision tree





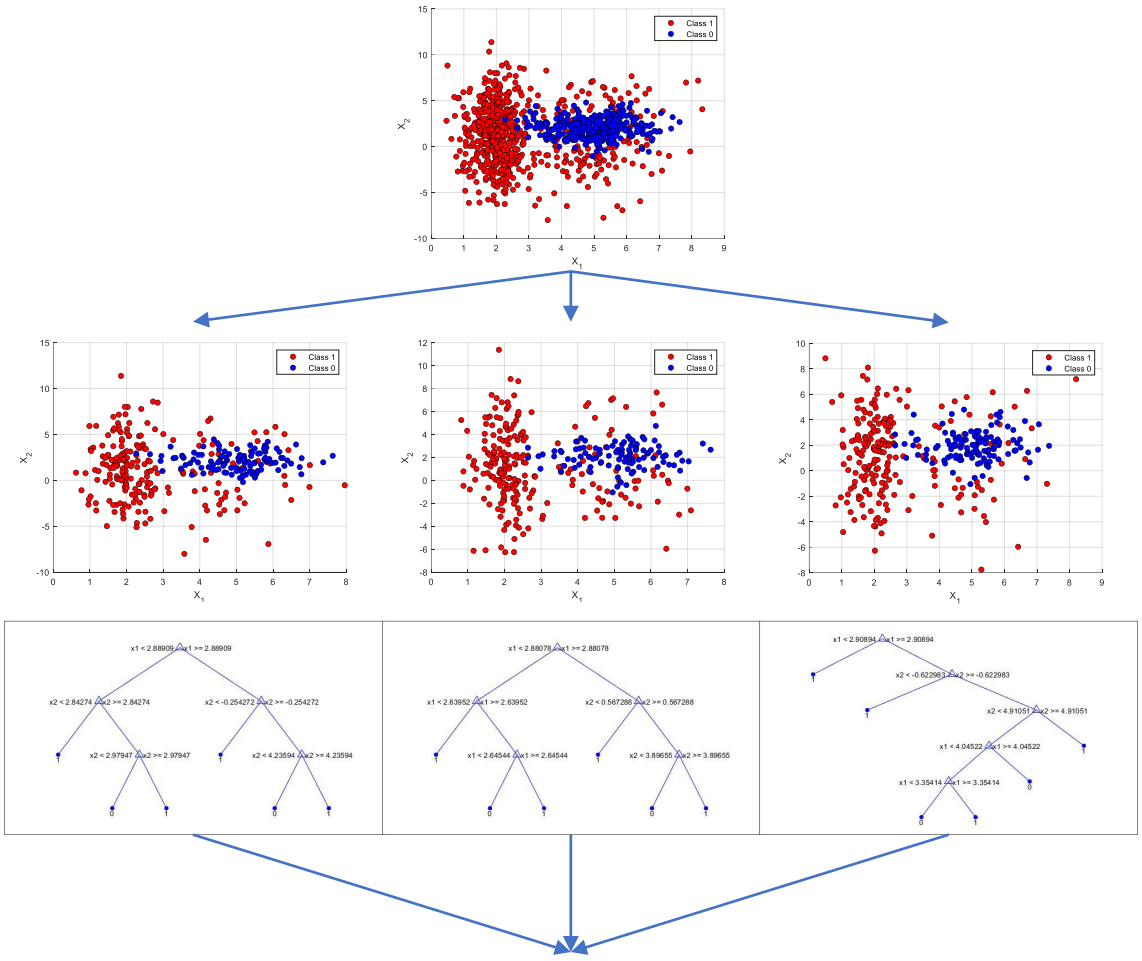
Ensemble Decision Trees

- Create multiple trees and aggregate the results.
- Ensemble methods usually deliver better performance at similar computational and algorithmic complexity.
- **Main types:**
  - Bagging
  - Random forest
  - Boosting

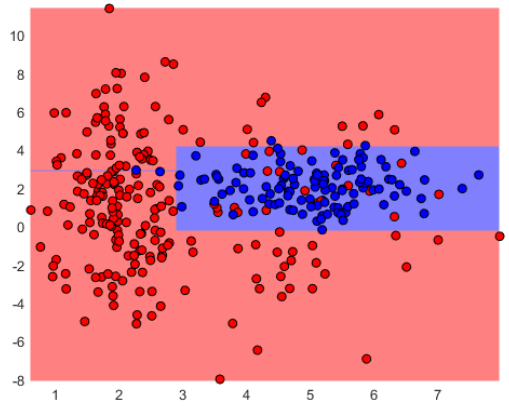
Individual trees can be constructed using parallel computations

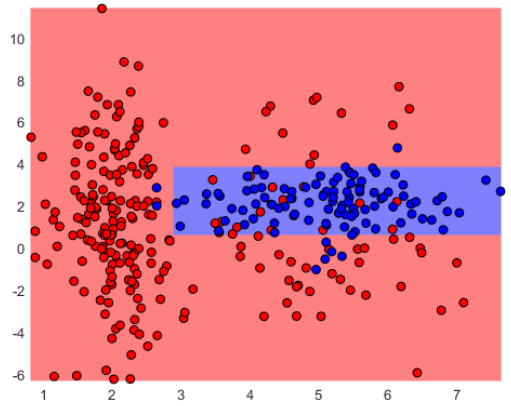
Bagging

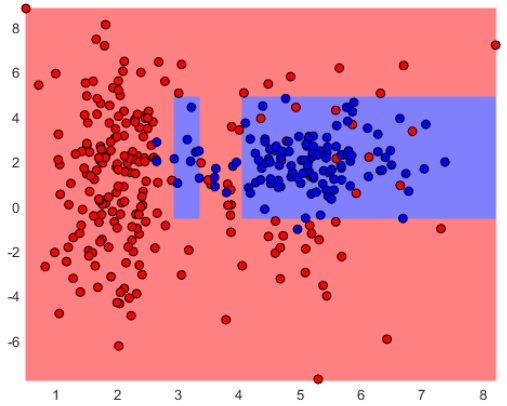
1. Split the total training data into a predetermined number of sets with random sampling with replacement.
2. Train decision tree for each dataset.
3. Aggregate results by averaging (for regression problem) or voting (for classification)

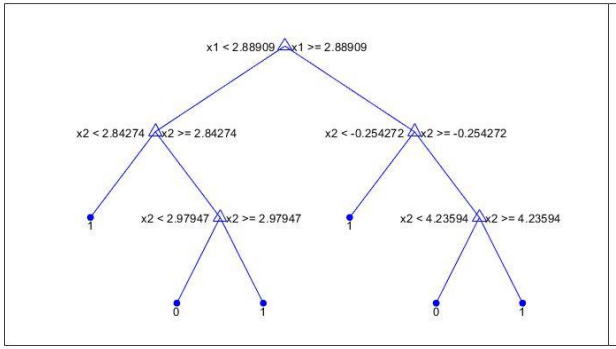


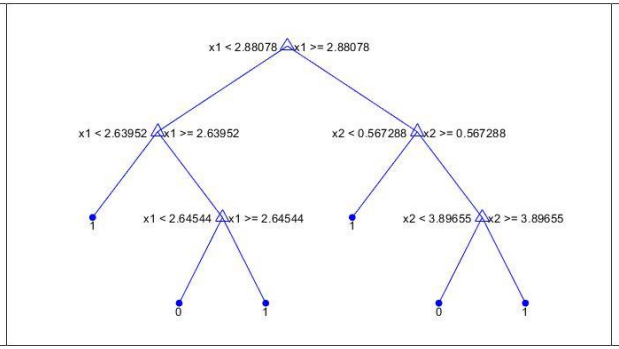
Bagging

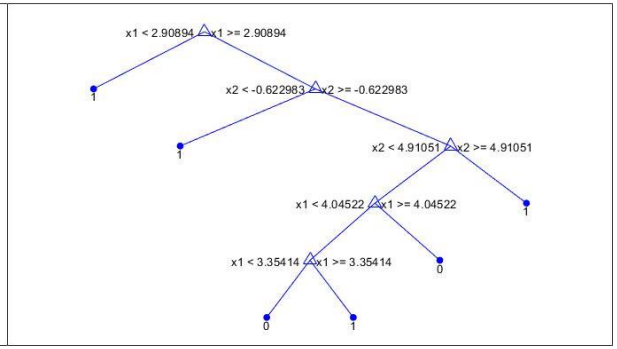

10
8
6
4
2
0
-2
-4
-6
-8
1
2
3
4
5
6
7


10
8
6
4
2
0
-2
-4
-6
1
2
3
4
5
6
7


8
6
4
2
0
-2
-4
-6
1
2
3
4
5
6
7
8


x1 < 2.88909
x1 >= 2.88909
x2 < 2.84274
x2 >= 2.84274
x2 < -0.254272
x2 >= -0.254272
x2 < 2.97947
x2 >= 2.97947
x2 < 4.23594
x2 >= 4.23594
1
0
1
0
1


x1 < 2.88078
x1 >= 2.88078
x1 < 2.63952
x1 >= 2.63952
x2 < 0.567288
x2 >= 0.567288
x1 < 2.64544
x1 >= 2.64544
x2 < 3.89655
x2 >= 3.89655
1
0
1
0
1


x1 < 2.90894
x1 >= 2.90894
x2 < -0.622983
x2 >= -0.622983
x2 < 4.91051
x2 >= 4.91051
x1 < 4.04522
x1 >= 4.04522
x1 < 3.35414
x1 >= 3.35414
1
1
1
0
0
1

Random Forest

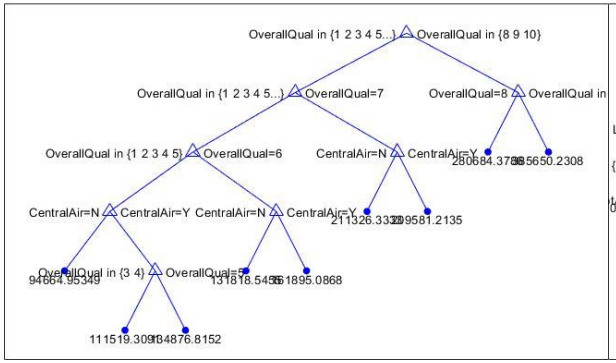
1. Split the total training data into a predetermined number of sets with random sampling with replacement.

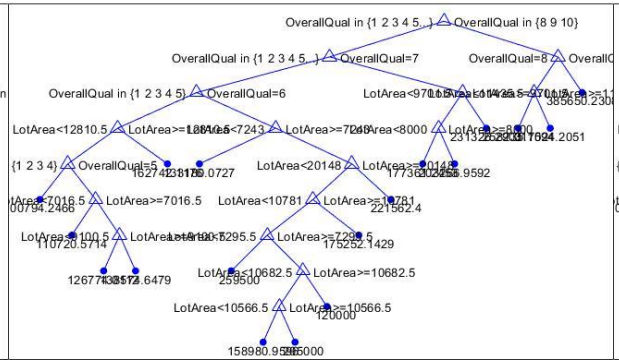
2. Train decision tree for each dataset and **random set of features**.

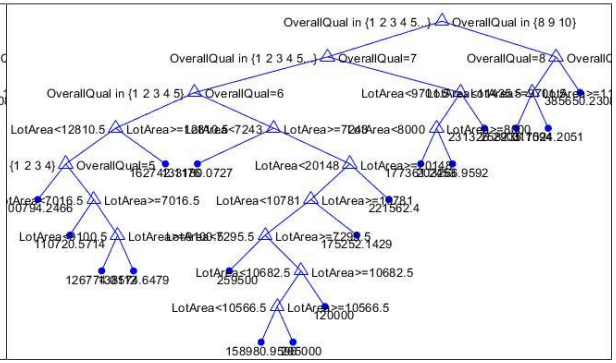
3. Aggregate results by averaging (for regression problem) or voting (for classification)

Id	LotArea	OverallQual	RoofStyle	CentralAir	Price
1	8450	7	Gable	Yes	150 000
2	9600	6	Gable	Yes	150 000
3	11250	7	Gable	Yes	80 000
4	9550	7	Mansard	No	150 000
5	14260	8	Gable	No	220 000
6	14115	5	Gable	Yes	85 000
7	10084	8	Gambrel	Yes	200 000
8	15245	7	Gable	Yes	180 000
9	6120	7	Gable	Yes	150 000
10	7420	4	Gable	Yes	50 000
…	…	…	…	…	…

Random Forest







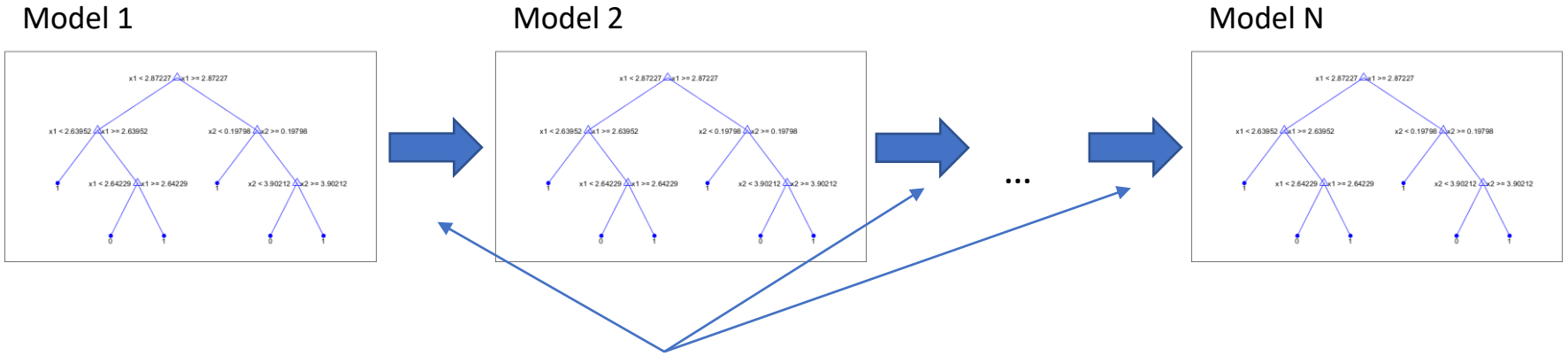
Features:
OverallQual
CentralAir

Features:
OverallQual
LotArea

Features:
OverallQual
LotArea
RoofStyle

Boosting

Models are constructed sequentially:



Evaluate misclassified examples in the previous model and assign them higher weight (importance) in loss function.

The aggregate result is a weighted aggregate of the predictions made by the individual models.

Example

- *decision_tree.m*
- *DMM.m*
- *train.csv*